BAB II

LANDASAN TEORI

2.1 Pengertian Hujan

Hujan adalah proses jatuhnya air dari awan yang terdapat di atmosfer bumi menuju permukaan bumi. Proses kondensasi (pemadatan) berperan dalam terciptanya hujan, kondensasi adalah proses pemadatan dari uap air menjadi butiran-butiran air. Butiran-butiran air yang terkumpul semakin lama akan semakin menjadi besar dan terbentuk menjadi awan, yang selanjutnya akan terbawa oleh angin yang akan menyebarkan hujan di permukaan bumi.

Butiran air yang jatuh memiliki beragam ukuran, yang berukuran lebih dari 0,5 mm akan menjadi hujan, lalu yang memiliki ukuran antara 0,2 mm sampai 0,5 mm akan menjadi gerimis, sedangkan yang berukuran lebih kecil dari itu akan menguap sebelum jatuh ke permukaan bumi (Irfan, dkk, 2005).

2.1.1 Proses Terjadinya Hujan

Terdapat 2 teori tentang bagaimana proses hujan bisa terjadi, yaitu:

1) Teori Kristal Es (*ice crystal theory*)

   Adalah hujan yang terbentuk dari butir-butir es yang terbentuk akibat butiran-butiran air yang tertarik oleh benih-benih es, dan terjadi pengembunan mendadak. Kristal es yang terlalu besar dan berat sehingga tidak mampu melayang lagi akan jatuh ke bumi, dalam perjalanan ke bumi mereka akan berubah menjadi air hujan karena pengaruh awan panas (Irfan, dkk, 2005).

2) Teori Tumbukan (*coalescene theory*)

   Dikarenakan oleh ukuran butiran air yang beragam, maka kecepatan jatuh mereka juga tidak akan sama. Karena perbedaan kecepatan ini, butiran yang memiliki ukuran yang berbeda akan saling bertumbukan dan dan menjadi ukuran yang lebih besar (Irfan, dkk, 2005).

2.1.2 Jenis-Jenis Hujan

Ada beberapa jenis hujan menurut (Novianta, dkk, 2011) yaitu:

Berdasarkan ukuran butiran, hujan dapat dibedakan menjadi:

1) Hujan gerimis, adalah hujan yang butiran airnya memiliki diameter kecil atau lebih tepatnya lebih kecil dari 0,5 mm.
2) Hujan salju, adalah hujan yang terbentuk dari kumpulan kristal es, kristal es ini terbentuk apabila suhu kurang dari 0°C.
3) Hujan batu es, adalah hujan batu es yang terjadi ketika suhu awan di bawah 0°C.
4) Hujan deras, adalah hujan yang memiliki diameter berukuran ± 7 mm.

Sedangkan berdasarkan proses terjadinya, hujan dibedakan menjadi (Novianta, dkk, 2011):

1) Hujan Frontal, adalah hujan yang terjadi akibat bertemunya massa udara panas dengan massa udara dingin.
2) Hujan Tropis, adalah hujan yang terjadi akibat dari suhu yang sangat tinggi, dan umumnya terjadi di daerah tropis.
3) Hujan Orografis, adalah hujan yang terjadi akibat udara yang mengandung uap air tertiup oleh angin sehingga mengarah ke atas lereng pegunungan, dimana semakin tinggi udara akan menjadi dingin, lalu terjadi pemadatan dan menjadi hujan.

2.2 Pengertian Curah Hujan

Curah hujan (mm) adalah ketinggian air hujan yang terkumpul dalam tempat yang datar, tidak menguap, tidak meresap dan tidak mengalir. Sebagai contoh curah hujan 1 mm berarti adalah air hujan yang memiliki ketinggian 1 mm yang tertampung pada tempat yang datar dengan luas 1 m^2 (Mulyono, 2014). Negara kita termasuk memiliki curah hujan yang cukup tinggi, hal ini dikarenakan kita berada di wilayah tropis.

2.2.1 Faktor yang Memengaruhi Curah Hujan

Beberapa faktor yang memengaruhi curah hujan seperti yang disebutkan oleh (Syah, 2015), adalah sebagai berikut:

1) Faktor garis lintang

   Perbedaan jumlah curah hujan yang disebabkan oleh hujan adalah apabila suatu daerah semakin dekat dengan garis lintang, maka kemungkinan dari curah hujan yang terjadi juga akan semakin tinggi di daerah tersebut. Hal ini dikarenakan di daerah yang bertempat di lintang rendah suhunya lebih besar daripada suhu di daerah yang bertempat di lintang tinggi, akibatnya penguapan juga akan semakin tinggi, dimana semakin terjadi penguapan maka semakin sering hujan akan terjadi.

2) Faktor ketinggian tempat

   Ketinggian suatu daerah dapat memengaruhi curah hujan dikarenakan semakin rendah suatu tempat akan memiliki suhu yang semakin tinggi, dan akibatnya akan memiliki curah hujan yang tinggi pula. Sebaliknya jika suatu daerah berada di tempat yang tinggi, maka curah hujannya akan rendah karena suhu yang rendah.

3) Jarak dari sumber air (penguapan)

   Hal ini disebabkan karena jika suatu daerah berlokasi dekat dengan sumber air (laut, sungai, danau dsb) maka kemungkinan terjadinya hujan juga akan semakin besar.

4) Deretan pegunungan

   Faktor pegungunan merujuk pada hujan orografis yang telah disebutkan di atas.

5) Perbedaan suhu daratan dan lautan

   Hujan akan lebih sering terjadi di daratan jika suhu daratan lebih rendah dari suhu laut dan berlaku sebaliknya.

6) Luas daratan

   Luas daratan yang semakin luas berpengaruh terhadap curah hujan, hal ini dikarenakan proses perjalan uap air dari sumber air yang semakin lama.

2.2.2 Pengukuran Curah Hujan

Curah hujan bisa diukur dengan menggunakan alat yang disebut dengan penakar hujan. Alat ini bekerja dengan mengukur tinggi air hujan yang jatuh ke tanah yang menumpuk di atas kolom air. Volume air lalu dibagi dengan luas corong penampung, hasilnya adalah tinggi atau tebal, untuk satuan yang

digunakan adalah milimeter (mm). Terdapat 3 jenis penakar hujan secara umum (Manullang dan Takdir, 2013): Penakar hujan biasa tipe *Obervatorium* atau konvensional, Penakar hujan mekanik *recorder* (Jenis Hellman), dan Penakar hujan otomatis *tipping bucket.*

2.3 Pengertian Prediksi

Prediksi adalah usaha untuk memerkirakan sesuatu yang akan terjadi di waktu mendatang dengan memanfaatkan berbagai informasi yang relevan pada waktu-waktu sebelumnya melalui suatu metode ilmiah. Prediksi juga memerkirakan besar atau jumlah sesuatu pada waktu yang akan datang berdasarkan data pada masa lampau yang dianalisis secara ilmiah (Hutabarat, dkk, 2018).

2.3.1 Tujuan Prediksi

Tujuan dari prediksi adalah untuk mengambil keputusan yang paling tepat berdasar pada pertimbangan dari masa lalu atau saat pengambilan keputusan tersebut, sehingga kita tidak akan mengalami masalah yang sama (Ginting dan Rosnani, 2007).

2.4 Pengertian Jaringan Saraf Tiruan

Jaringan saraf tiruan adalah *prosesor* yang terdistribusi besar-besaran secara paralel yang dibuat dari unit proses sederhana, dengan kemampuan untuk menyimpan pengetahuan berupa pengalaman dan dapat digunakan untuk proses lain (Haykin, 2009).

Lalu menurut (Hutabarat, dkk, 2018), dijelaskan bahwa jaringan saraf tiruan merupakan otak buatan manusia yang diimplementasikan menggunakan program komputer untuk mensimulasikan proses pembelajaran.

Sedangkan menurut (Jumarwanto, dkk, 2009), bahwa jaringan saraf tiruan merupakan suatu sistem pemrosesan informasi yang mempunyai sifat dan karakteristik menyerupai jaringan saraf manusia.

Dari dua pengertian di atas bisa ditarik kesimpulan tentang definisi dari jaringan saraf tiruan, yaitu sebuah sistem pemrosesan informasi yang mampu meniru cara kerja sistem jaringan saraf manusia. Sistem ini mampu belajar dari pengalaman dan menarik sebuah keputusan untuk kedepannya.

Jaringan saraf tiruan dibentuk sebagai generalisasi model matematika dari jaringan saraf biologis manusia, dengan asumsi bahwa (Harto, dkk, 2006):

1) Pemrosesan informasi terjadi pada banyak elemen sederhana (*neuron*).
2) Sinyal dikirimkan di antara *neuron-neuron* melalui penghubung-penghubung.
3) Penghubung antar *neuron* membawa bobot yang akan berpengaruh dalam menguatkan atau melemahkan sinyal.
4) Untuk menentukan nilai *output*, setiap *neuron* akan menggunakan fungsi aktivasi yang dikenakan pada jumlah *input* yang diterima. Besarnya *output* ini selanjutnya akan dibandingkan dengan suatu batas ambang.

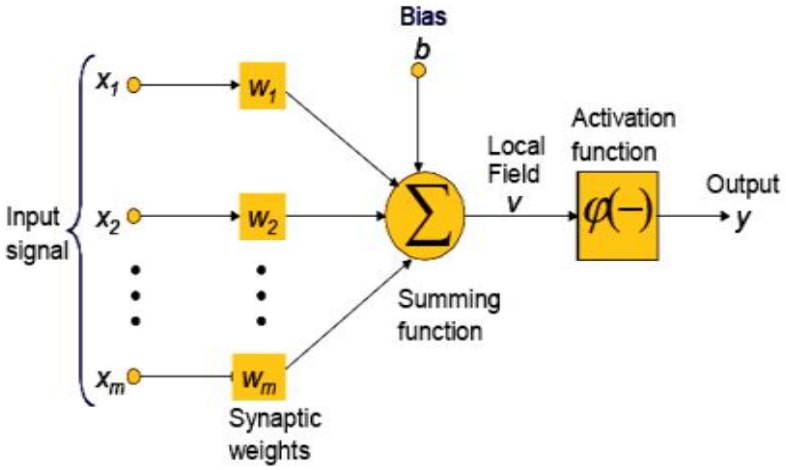
Ada dua kesamaan jaringan saraf tiruan dengan cara kerja otak manusia, yaitu yang pertama pengetahuan diperoleh jaringan melalui proses belajar. Kemudian yang kedua kekuatan hubungan antar *neuron* yang dikenal sebagai bobot-bobot sinaptik digunakan untuk menyimpan pengetahuan.

2.4.1 Prinsip Jaringan Saraf Tiruan

Terdapat 3 prinsip jaringan saraf tiruan menurut (Siang, 2004):

1) Arsitektur jaringan atau pola hubungan antar *neuron.*
2) *Training* dan *learning* atau penentuan bobot.
3) Fungsi aktivasi, yaitu berfungsi untuk menentukan *output* dari suatu *neuron*.

Gambaran sederhana dari prinsip jaringan saraf tiruan:



Gambar 2.1 Prinsip Dasar Jaringan Saraf Tiruan (Jumarwanto, dkk, 2009)

Pada gambar di atas, y menerima masukan dari *neuron* x_1, x_2, dan x_3, dengan bobot hubungan masing-masing adalah w_1, w_2, dan w_3. Ketiga impuls *neuron* yang ada dijumlahkan menjadi:

$$\boldsymbol{Net = x_1 w_1 + x_2 w_2 + x_3 w_3} \qquad (2.1)$$

Besarnya impuls yang diterima oleh y mengikuti fungsi aktivasi $y = f(net)$. Apabila nilai fungsi aktivasi cukup kuat, maka sinyal akan diteruskan. Nilai dari *output* akan digunakan untuk mengubah bobot.

2.4.2 Konsep Dasar Jaringan Saraf Tiruan

Nilai dari *input* dan *output* akan diproses di dalam *neuron*, dan *neuron* berkumpul di dalam lapisan yang disebut *neuron layers*. Untuk lapisan *neuron* tersebut ada 3, yaitu (Agustin, 2012):

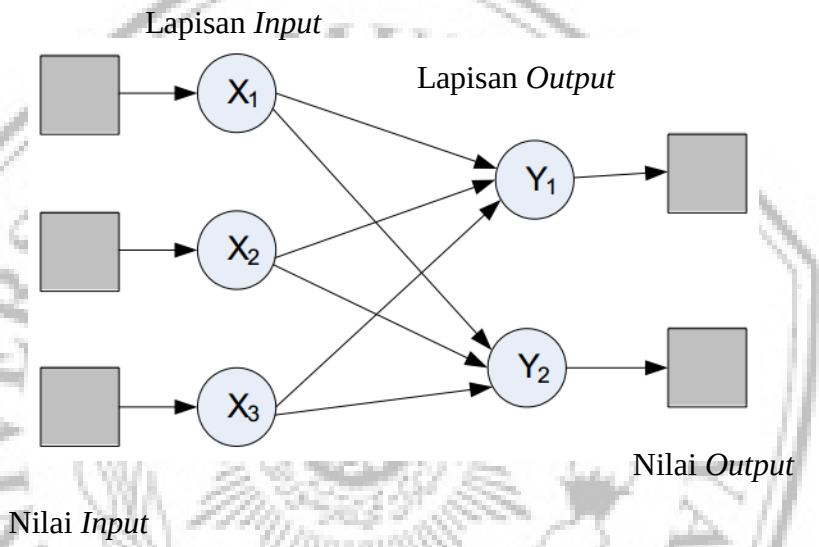
1) *Input layer*, adalah lapisan yang berisi unit-unit masukan yang menggambarkan suatu permasalahan dari luar.
2) *Hidden layer*, adalah lapisan untuk memroses unit *input* sebelum menuju lapisan *output.*
3) *Output layer*, adalah lapisan yang berisi solusi atau hasil dari pemrosesan dari suatu permasalahan.

2.4.3 Arsitektur Jaringan Saraf Tiruan

Terdapat 3 jenis arsitektur jaringan saraf tiruan menurut (Agustin, 2012):

1) *Single layer network*

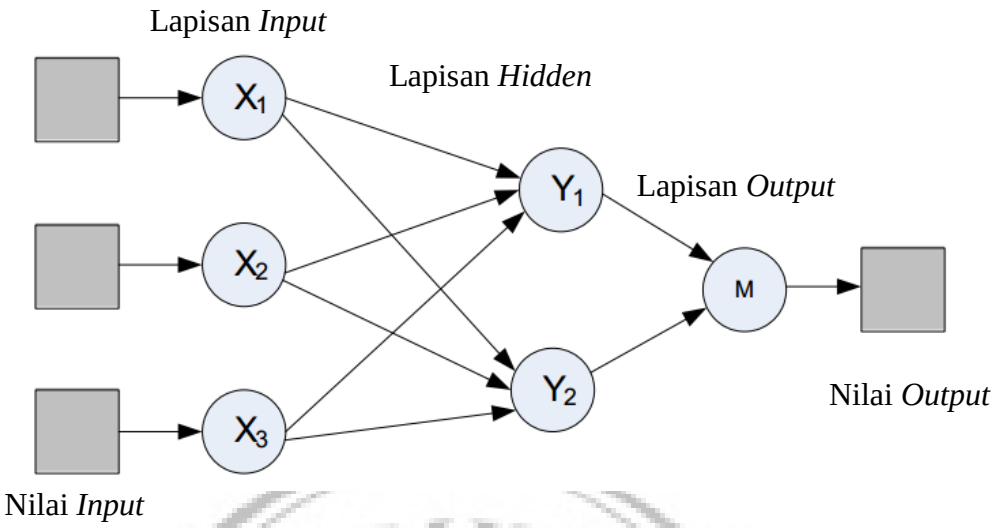
Jaringan yang hanya memiliki satu lapisan yang terdiri dari 1 *input layer* dan 1 *output layer*. Seluruh unit yang berada di dalam *input layer* akan selalu terhubung dengan *output layer*. Proses yang dilakukan dengan arsitektur ini adalah secara langsung mengolah *input* untuk kemudian menjadi *output*. Beberapa contoh algoritma yang menggunakan arsitektur ini adalah: *Perceptron, Hopfield,* dan *ADALINE*.



Gambar 2.2 Arsitektur *Single Layer Network* (Agustin, 2012)

2) *Multi layer network*

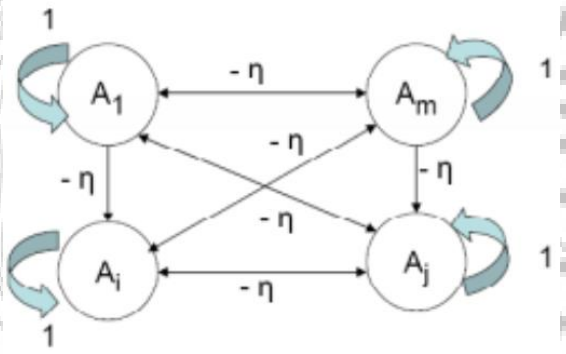
Arsitektur ini memiliki 3 lapisan yaitu *input layer*, *hidden layer*, dan *output layer*. Arsitektur ini mampu menyelesaikan masalah yang lebih rumit jika dibanding dengan *single layer network*, tetapi biasanya dengan konsekuensi proses yang lebih lama. Beberapa contoh algoritma yang menggunakan arsitektur ini adalah: *backpropagation, neocognitron*, dan *MADALINE*.



Gambar 2.3 Arsitektur *Multi Layer Network* (Agustin, 2012)

3) *Competitive layer network.*

Sekumpulan *neuron* bersaing untuk mendapatkan hak menjadi aktif di dalam jaringan ini. Contoh algoritma yang menggunakan arsitektur ini adalah: LVQ.



Gambar 2.4 Arsitektur *Competitive Layer* (Agustin, 2012)

2.4.4 Metode Pelatihan Jaringan Saraf Tiruan

Terdapat 3 metode dari pelatihan jaringan saraf tiruan menurut (Agustin, 2012):

1) *Supervised Learning*

Selisih (*error*) dari pola *output* aktual dengan *output* target (yang diinginkan) digunakan untuk mengubah bobot dengan tujuan untuk menghasilkan *output* aktual yang sedekat mungkin dengan *output* target. Contoh algoritma yang menggunakan metode ini adalah: *Backpropagation*, *ADALINE*, dan *Perceptron.*

2) *Unsupervised Learning*

Dengan metode ini kita tidak dapat menentukan *output* target, kita hanya bisa menyusun nilai bobot dalam *range* tertentu tergantung dari *input* yang diberikan. Tujuan dari metode ini mengelompokkan unit-unit yang hampir sama dalam suatu area tertentu. Contoh algoritma menggunakan metode ini adalah: *Neocognitron, LVQ,* dan *Competitive.*

3) *Hybrid Learning*

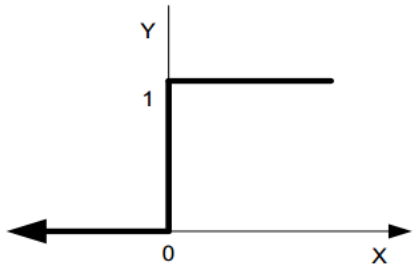
Adalah metode penggabungan dari dua metode di atas, penentuan bobot sebagian dilakukan melalui *supervised learning* dan sebagian menggunakan *unsupervised learning.*

2.4.5 Fungsi Aktivasi Jaringan Saraf Tiruan

Fungsi aktivasi berperan dalam menentukan *output* dari suatu *neuron* (Agustin, 2012). Ada 3 fungsi aktivasi yang umum dipakai adalah:

a) Fungsi *Threshold*

Fungsi *threshold* merupakan fungsi *threshold* biner. Untuk kasus yang memiliki bilangan bipolar, maka angka 0 diganti dengan angka -1. Dalam jaringan saraf tiruan terkadang ditambahkan suatu unit *input* yang selalu bernilai 1. Unit tersebut disebut dengan bias. Bias berfungsi untuk mengubah *threshold* menjadi = 0.



Gambar 2.5 Fungsi aktivasi *Threshold* (Agustin, 2012)

$$F(x) = \begin{cases} 1 \\ 2 \end{cases} \qquad (2.2)$$

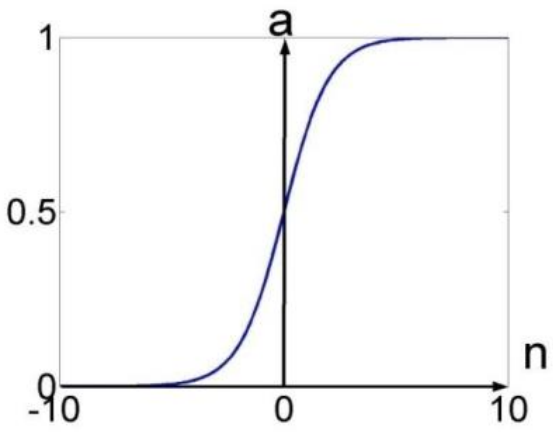
Jika $x \geq a$

Jika $x < a$

b) Fungsi *Sigmoid*

Fungsi ini cukup sering digunakan karena kemudahannya dalam mendiferensiasikan nilai fungsinya.

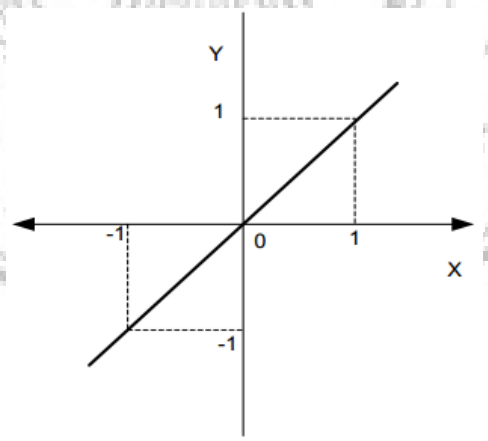
$$F(x) = \frac{1}{1+e^{-x}} \qquad (2.3)$$



Gambar 2.6 Fungsi aktivasi *Sigmoid* (Agustin, 2012)

c) Fungsi Identitas

Fungsi ini kita pakai ketika *output* yang dihasilkan oleh jaringan saraf tiruan merupakan sembarang bilangan riil (bukan hanya pada *range* [0,1] atau [1,-1]). $X = Y$



Gambar 2.7 Fungsi aktivasi Identitas (Agustin, 2012)

2.5 Jaringan Saraf Tiruan *Backpropagation*

Algoritma jaringan saraf tiruan *backpropagation* yang termasuk dalam *supervised learning* ini memiliki kemampuan penanganan masalah pengenalan pola yang rumit.

Memiliki arsitektur dimana unit yang terdapat di dalam *input layer* terhubung dengan unit yang berada di dalam *hidden layer*, dan unit di dalam *hidden layer* terhubung dengan unit di dalam *output layer*.

Cara kerja dari algoritma ini adalah pola *input* akan diproses di dalam *hidden layer*, lalu diteruskan menuju unit-unit di dalam *output layer*. Apabila hasil *output* tidak sesuai target, maka akan dikirim mundur (*backward*) menuju *hidden layer* dan kembali ke *input layer* (Agustin, 2012).

Proses di atas dilakukan untuk melatih suatu jaringan saraf tiruan, yaitu dengan cara melakukan perubahan bobot, sedangkan penyelesaian masalah akan dilakukan jika proses pelatihan tersebut telah selesai, fase ini disebut dengan pengujian (Agustin, 2012).

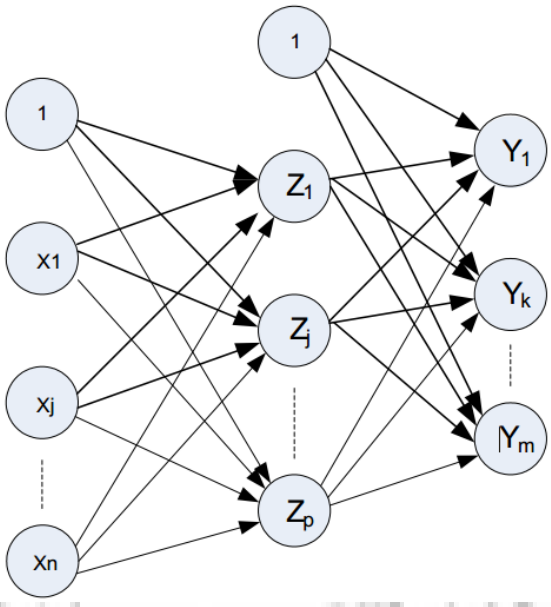
Sedangkan menurut (Hutabarat, dkk, 2018), algoritma *backpropagation* merupakan algoritma iteratif sederhana yang memiliki sifat komputasi yang baik dan mampu bekerja dengan baik dengan data yang berskala besar dan kompleks.

Jadi bisa diambil kesimpulan mengenai jaringan saraf tiruan *backpropagation*, bahwa metode ini bekerja dengan cara meminimalisir tingkat *error* dengan cara mengubah nilai bobot sesuai dengan selisih antara nilai *output* dan target yang diinginkan.

2.5.1 Arsitektur Jaringan Saraf *Backpropagation*

Arsitektur dari jaringan saraf *backpropagation* terdiri dari 3 *layer* (Agustin, 2012) yaitu:

1) *Input layer* (1 buah), terdiri dari 1 hingga n unit *input*.
2) *Hidden layer* (minimal 1 buah), terdiri dari 1 hingga p unit tersembunyi.
3) *Output layer* (1 buah), terdiri dari 1 hingga m unit *output*.



Gambar 2.8 Arsitektur Jaringan *Backpropagation* (Agustin, 2012)

2.5.2 Pelatihan Jaringan Saraf *Backpropagation*

Ada 2 tahap pelatihan dari jaringan saraf *backpropagation* yaitu *feedforward* dan *backward propagation*. Jaringan akan diberikan set pelatihan yang terdiri dari vektor *input* dan vektor *output* target. Perbandingan dari *output* aktual dan *output* target yang disebut dengan *error* akan dijadikan sebagai dasar untuk mengubah bobot. *Error* akan berkurang setiap kali bobot diubah, proses pengubahan bobot akan terus dilakukan pada setiap set pelatihan dan akan berhenti ketika sudah mencapai nilai ambang yang ditetapkan atau ketika mencapai batas iterasi yang telah ditetapkan (Agustin, 2012).

Tahapan pelatihan jaringan saraf tiruan *backpropagation* (Agustin, 2012):

1) Tahap *feedforward*
2) Tahap *backward propagation*
3) Tahap pengubahan bobot dan bias.

   Langkah 0: Inisialisasi bobot-bobot, konstanta laju pelatihan (α), toleransi *error* atau nilai bobot atau set maksimal *epoch*/iterasi

   Langkah 1: Mengulang langkah-2 hingga langkah ke-9 sampai syarat berhenti belum tercapai.

   Langkah 2: Untuk setiap pasangan pola pelatihan, lakukan langkah ke-3 sampai langkah ke-8.

Langkah 3: Tiap unit *input* menerima sinyal dan meneruskannya ke unit *hidden* diatasnya.

Langkah 4: Tiap unit di *hidden layer* (dari unit ke-1 hingga unit ke-p) dikalikan dengan bobotnya dan dijumlahkan serta ditambahkan dengan biasnya.

Langkah 5: Tiap unit *output* ($y_k, k = 1,2,3, \dots m$) dikalikan dengan bobot dan dijumlahkan serta ditambahkan dengan biasnya.

Langkah 6: Masing-masing unit *output* ($y_k, k = 1,2,3, \dots m$) menerima pola target t_k sesuai dengan pola *input* saat pelatihan dan kemudian informasi *error* lapisan *output* (δk) dihitung. δk dikirim ke lapisan di bawahnya dan digunakan untuk menghitung besarnya koreksi bobot dan bias (ΔW_{jk} dan ΔW_{ok}) antara *hidden layer* dengan *output layer*.

Langkah 7: Tiap unit di *hidden layer* (dari unit ke-1 hingga ke-p; $i = 1 \dots n; k = 1 \dots m$) dilakukan perhitungan informasi kesalahan *hidden layer* (δ_j). δ_j kemudian digunakan untuk menghitung besar koreksi bobot dan bias (ΔV_{ji} dan ΔV_{jo}) antara lapisan *input* dan *hidden layer*.

Langkah 8: Tiap unit *output* ($y_k, k = 1,2,3, \dots m$) dilakukan pengubahan bias dan bobotnya ($j = 0,1,2, \dots p$) sehingga menghasilkan bobot dan bias baru. Demikian juga untuk setiap unit *hidden* mulai dari unit ke-1 sampai dengan unit ke-p dilakukan pengubahan bobot dan bias.

Langkah 9: Uji kondisi berhenti (akhir iterasi).

2.5.3 Optimalisasi Jaringan Saraf Tiruan *Backpropagation*

Menurut (Jumarwanto, 2009), bahwa terdapat beberapa cara untuk mengoptimalkan JST *backpropagation*, yakni diantaranya:

1) Pemilihan bobot awal bias

   Bobot awal sangat berpengaruh terhadap apakah jaringan bisa mencapai titik global atau lokal, dan juga memengaruhi kecepatan konvergensi. Pemilihan

nilai bobot awal sedapat mungkin tidak yang terlalu besar atau terlalu kecil, karena akan berpengaruh pada nilai turunan fungsi aktivasi.

2) Jumlah *hidden unit*

   Meskipun algoritma *backpropagation* yang hanya menggunakan satu *hidden layer* bisa dibilang sudah cukup untuk mendapatkan target *output* yang diinginkan, namun dengan menambahkan jumlah *hidden layer* tentu akan membuat pelatihan yang lebih optimal.

3) Jumlah pelatihan

   Jumlah pola pelatihan dipengaruhi oleh banyaknya bobot dan tingkat akurasi yang dikehendaki.

4) Lama iterasi

   Jumlah iterasi tidak akan terlalu berpengaruh untuk mendapatkan pola pengujian yang tepat, hal ini dikarenakan jaringan *backpropagation* bertujuan untuk mendapatkan keseimbangan antara pengenalan pola pelatihan yang benar dan respon yang baik untuk pola lain sejenis lainnya.

5) Parameter *learning rate*

   Learning rate atau laju pelatihan memengaruhi proses kecepatan dan efektivitas pelatihan. Pemilihan nilai yang tepat dari *learning rate* tergantung pada masalah yang ini diselesaikan. Nilai *learning rate* yang kecil dapat menjamin penurunan *gradient* akan berjalan dengan baik, tetapi dengan konsekuensi jumlah iterasi yang bertambah.